Challenges of cryptographic engineering of non-binary pseudorandom sequence generators

ITEFI, CSIC, Madrid, 22.05.2024

Slobodan Petrović, NTNU/IIK, Norway

Motivation

- Limitations of binary electronics
  - Moore's curse – 3 scaling walls
    - Power wall
      - Problems with delivering and dissipation of power as density of transistors increases
      - Relatively long wiring inside the chip also generates heat
    - Memory wall
      - Latency and bandwidth gap between on-chip data processing and off-chip data retrieval
      - Processors' power has been growing exponentially, while memory performance increased linearly – this has now stopped (after 2005), however the gap is huge, thus almost no benefit with increasing processing power of CPU while memory retrieval is so inefficient
    - Electronic Design Automation (EDA) wall
      - Chip design, simulation, and verification tools development cannot follow the technology development fast enough

Motivation

- Post-binary electronics – CMOS replaced by
  - CNTFET (Carbon Nanotube FET) – as cheap as binary technology!
  - Memristor – experimental, but promising
  - TCMOS (Ternary CMOS), etc.
- Three-state logic considered most often
  - Multiple-Valued Logic (MVL) with more than 3 levels also intensively studied
  - Still, ternary is considered most practical, commercial hardware exists
- Historically, first ternary computers were built in mid-20th century in USSR (Setun, Setun-70), but they were not widely accepted – the world trend was binary

Motivation

- The radix economy – a cost metric to compute the optimal radix
  - rw – product
    - r – the radix (modulus)
    - w – number of positions needed to encode a random integer $n \in \{0, \dots, N-1\}$
    - Example – to encode 5 in binary, we need $w = 3$ positions ($r = 2$, so $rw = 6$), in unbalanced ternary (the symbols are $0, 1, 2$) we need $w = 2$ positions ($r = 3$, so $rw = 6$), in balanced ternary (the symbols are $-1, 0, 1$) we need $w = 3$ positions ($r = 3$, so $rw = 9$)
      - Binary: 101
      - Unbalanced ternary: 12
      - Balanced ternary: $1, -1, -1$ (+-- or $1ZZ$ or $1TT$) ($= 1 \times 3^2 - 1 \times 3^1 - 1 \times 3^0$)
    - Averaged the function rw over all $n \in \{0, \dots, N-1\}$, it turns out that the minimum of this average is obtained for $r \approx e$, which is closer to 3 than to 2

Motivation – 7 Cs of benefit

- IoT applications – a promising environment for ternary logic (1)
  - Computation
    - Greater “digit-size” of a device can be achieved with a smaller number of “transistors”
  - Communication
    - MVL is natural for communication
      - WiFi 7 (IEEE 802.11be) uses QAM-4096 (base 12)
      - Bluetooth classic uses 8-DPSK (base 8)
      - USB 4.2 uses binary encoded ternary (PAM-3)
  - Consumption of energy
    - On average, less power consumed on state transitions in ternary than in binary
    - Possible to make energy-efficient MVL circuits with CNTFETs and TCMOS

Motivation – 7 Cs of benefit

- IoT applications – a promising environment for ternary logic (2)
  - Compression
    - A smaller number of ternary memory devices needed to store the same amount of information than in binary
    - Example (trit – tryte in ternary analogous to bit-byte in binary)
      - Setun's tryte had 6 trits (like with bytes, CDC byte had 6 bits, IBM byte had 8 bits)
      - 1 tryte = 8 trits $= \log_2 3^8 \approx 12.65$ bits
      - 1 tryte = 6 trits $= \log_2 3^6 \approx 9.5$ bits
      - 1 tryte = 5 trits $= \log_2 3^5 \approx 7.9$ bits, similar to binary – a motivation for use of a 5-trit tryte
  - Comprehension
    - With ternary, we can model the processes that include partial or unknown information, not only those that are well-modelled with true or false only (Kleene logic)

Motivation – 7 Cs of benefit

- IoT applications – a promising environment for ternary logic (3)
  - Cybersecurity
    - More information density offered in ternary enables larger key spaces with the same number of units (e.g., 128-trit keys instead of 128-bit keys)
    - Ternary Physical Unclonable Functions (PUFs) – such circuits already exist with CNTFETs
    - By mixing binary and ternary signals in encryption schemes the logic states are obscured, and it is difficult to follow the signals by means of binary side-channel attack hardware
    - Differential Power Analysis (DPA) also more difficult if MVL is used for implementation of cryptographic solutions
  - Complexity
    - With 2 inputs and 1 output, there are 2^4 possible Boolean functions in binary, while in ternary, with 2 inputs and 1 output, we can realize 3^9 functions

Ternary computing in practice

- Balanced ternary most often used
  - Negative numbers easily represented and realized (3's complement not necessary)
  - Easier to realize in electronic circuits than unbalanced ternary
  - Easier to combine with binary circuits
- Trit instead of a bit
- Tryte instead of a byte
  - Most often 6 trits (like Setun), but also 3, 5, or 9 trits have been used
- Heptavintimal (base 27) instead of hexadecimal (or octal)

Ternary computing in practice

Decimal	0	1	2	3	4	5	6	7	8	9	10	11	12	13
Ternary	000	001	002	010	011	012	020	021	022	100	101	102	110	111
Heptavintimal	0	1	2	3	4	5	6	7	8	9	A	B	C	D

Decimal	14	15	16	17	18	19	20	21	22	23	24	25	26
Ternary	112	120	121	122	200	201	202	210	211	212	220	221	222
Heptavintimal	E	F	G	H	K	M	N	P	R	T	V	X	Z

Infrastructure for cryptodesign in MVL

- Pseudorandom sequence generators realized in MVL technology
  - Basic building block – Linear Feedback Shift Register (LFSR)
    - On the level of the whole period – satisfies the 3 Golomb's postulates if the feedback polynomial is primitive
  - Another important building block – Boolean function
    - Non-linear filter (input – several stages from the same LFSR)
    - Non-linear combiner (input – outputs of several LFSRs)
  - Typical pseudorandom sequence generator for IoT based on LFSRs
    - A single LFSR of moderate length (up to 100 bits/trits)
    - A non-linear filter function
  - In MVL, to realize LFSR, we need primitive polynomials in GF(p), $p > 2$

Infrastructure for cryptodesign in MVL

- Realizing ternary gates in practice has been solved
- Realizing memory (flip-flops) in ternary has been a challenge so far
  - They are needed to realize ternary LFSR in practice
- Recently, ternary shift registers with CNTFETs have been produced
- For example
  - Yamani et al., Design an energy efficient pulse triggered ternary flip flops with Pseudo NCFET logic, Analog Integrated Circuits and Signal Processing (2024) 119:151–163

Infrastructure for cryptodesign in MVL

- There exists software that can help in finding primitive polynomials in large extension Galois fields with a small non-binary ground field
  - MATLAB has a coding-theory package implementing some algorithms that can be useful for this
  - Open-source – for example, Sage (www.sagemath.org) - demo
- But they do not offer a direct answer to a typical question that we ask in cryptodesign
  - We need m primitive polynomials of a given degree n with k non-zero feedback coefficients in GF(p)
- That is the reason why we build our own infrastructure - demo

Infrastructure for cryptodesign in MVL

- Primitive polynomial
  - The order of a polynomial $P(x)$, $\deg P(x) = n$, $P(0) \neq 0$ is the smallest integer e for which $P(x)$ divides $x^e - 1$
  - In GF(p^n), if the order of an irreducible polynomial $P(x)$ is $p^n - 1$, this polynomial is called primitive polynomial
  - To test whether a polynomial $P(x)$, $\deg P(x) = n$ in GF(p^n) is primitive
    - Test whether $P(x)$ is irreducible
    - If $P(x)$ is irreducible, check whether it divides the polynomials $x^k - 1, n \leq k < p^n - 1$
    - If $P(x)$ does not divide any of the polynomials above, then it is primitive
  - Obviously, this procedure is not efficient

Infrastructure for cryptodesign in MVL

- Primitive polynomial
  - Theorem (Alanen, Knuth, 1964; Herlestam, 1982)
    - A polynomial $f(x)$ in GF(q), $q = p^n$, $\deg f(x) = n$, is primitive if and only if it satisfies the following
      1. $\forall x \in \mathrm{GF}(q), f(x) \neq 0$
      2. $x^{p^n} \equiv x \pmod{f(x)}$
      3. For all prime factors p' of $p^n - 1$

         $x^{(p^n-1)/p'} \not\equiv 1 \pmod{f(x)}$
  - If $f(x)$ is irreducible, then the conditions 1. and 2. are satisfied
  - The condition 3. is trivially satisfied if $p^n - 1$ is a prime
    - $p^n - 1$ can be a prime if $p = 2$ (Mersenne primes), but not for $p > 2$ (then, obviously, $p^n - 1$ always has the factor 2)

Infrastructure for cryptodesign in MVL

- Primitive polynomial – practical test
  - Test the polynomial $f(x)$ for irreducibility
  - If $f(x)$ is irreducible, then test the condition 3. of the Alanen-Knuth-Herlestam's theorem
- How to do this in practice in a large extension field with a non-binary ground field?

Infrastructure for cryptodesign in MVL

- Testing irreducibility efficiently
  - Theorem
    - If a polynomial $f(x)$ of degree n with coefficients in GF(p) does not have common factors with

$$\left(x^{p^k} - x\right) \bmod f(x), 1 \leq k \leq \left[\frac{n}{2}\right]$$

    then it is irreducible
  - Computing $x^{p^k} - x$ efficiently, for a (relatively) large k
    - Solution – modular exponentiation
      - Raising x to the power of p and reducing modulo $f(x)$, for $1 \leq k \leq \left[\frac{n}{2}\right]$
  - gcd($\left(x^{p^k} - x\right) \bmod f(x), f(x)$) – Euclidean algorithm for polynomials

Infrastructure for cryptodesign in MVL

- Testing the condition 3. of the Alanen-Knuth-Herlestam's theorem
  - In general, we need factorizations of the integers, whose form is $p^n - 1$
  - For IoT, the interesting values for n are < 1000
    - In binary, there are factoring algorithms that proved to be very successful in factoring integers of the form $2^n - 1$ (e.g., Lucas-Lehmer)
      - In addition, there are many Mersenne primes in this range
        - for $n = 2, 3, 5, 7, 13, 17, 19, 31, 61, 89, 107, 127$ etc.
      - For these n, a polynomial is primitive if it is irreducible
    - For $p > 2$, it is more difficult, but much has been achieved regarding factoring
      - Tables of factorizations exist
        - For example, Brillhart et al., 1988 – factors of $3^n - 1$, $5^n - 1$, $7^n - 1$

Infrastructure for cryptodesign in MVL

- Practical realization
  - We assume that, given n, the factorization of $p^n - 1$ is known
  - Suppose $p^n - 1 = p_1 p_2 \cdots p_r$ (multiplicity does not matter, we check only the quotients $(p^n - 1)/p_i$)
  - We pre-compute these quotients and encode them in binary
  - Then we use modular exponentiation to compute $x^{(p^n-1)/p_i} \bmod f(x)$
  - Demo – show the factorizations of $2^n - 1$, $3^n - 1$, $5^n - 1$ up to $n = 257$

Infrastructure for cryptodesign in MVL

- Implementation
  - We must implement basic operations with polynomials (simplifications that hold in GF(2) do not hold for $p > 2$)
    - Addition, subtraction
    - Multiplication, division
    - gcd for polynomials
    - Powering of a polynomial (modular exponentiation)
    - LFSR synthesis (the Berlekamp-Massey algorithm)
  - Many small auxiliary routines that are not needed in binary
    - Example – conversion to a monic polynomial, conversion to a polynomial with positive coefficients etc.

Infrastructure for cryptodesign in MVL

- Boolean functions (1)
  - For stream ciphers, we need balanced, non-linear functions, far away from the linear ones
    - Bent functions are “the most distant from linear”, but unbalanced
    - We have to find a compromise – as non-linear and distant from linear as possible, but balanced
  - How do we find such functions?
    - In binary, we can compute Walsh transform to check how far the given function is from linear functions – for bent functions, all the values are $\pm 2^{n/2}$
    - In ternary (MVL in general) we can use the generalized Walsh transform
      - Vilenkin-Chrestenson transform – has been used to obtain ternary bent functions
        - For ternary bent functions, all the values are $\pm 3^{n/2}$

Infrastructure for cryptodesign in MVL

- Boolean functions (2)
  - In binary, given a Boolean function, we can compute its Walsh transform
    - If the values of the Walsh transform are relatively small, the function is far from linear
  - In MVL (including ternary), we can compute Vilenkin-Chrestenson transform and perform a similar check
  - Checking balancedness is easy – through the truth table
  - Practical procedure
    - Generate a (ternary) Boolean function at random
    - Check its Vilenkin-Chrestenson spectrum
    - Check the balancedness by inspecting the truth table

Infrastructure for cryptodesign in MVL

- Boolean functions (3)
  - Practical realization
    - In binary
      - Algebraic Normal Form (ANF) from the truth table
      - Walsh transform
    - In ternary (in progress)
      - Algebraic Normal Form (ANF) from the truth table
      - Vilenkin-Chrestenson transform

Resources (1)

- Brousentsov N. P., Maslov S. P., Ramil Alvarez J., Zhogolev E.A., Development of ternary computers at Moscow State University, https://www.computer-museum.ru/english/setun.htm
- Ternary research group at University of South-Eastern Norway https://ternaryresearch.com/
- McEliece R. J., Finite Fields for Computer Scientists and Engineers, Kluwer, 1987
- Brillhart J., Lehmer D.H., Selfridge J.L., Tuckerman B., Wagstaff S.S. junior, Factorizations of $b^n \pm 1$ Up to High Powers, AMS, 1988

Resources (2)

- Bos S., PhD thesis, University of South-East Norway, May 2024, https://openarchive.usn.no/usn-xmlui/handle/11250/3127984
- Thornton M.A., Modeling Digital Switching Circuits with Linear Algebra, re-print, Springer, 2022
- Stanković S. et al., Representation of Multiple-valued Bent Functions Using Vilenkin-Chrestenson Decision Diagrams, 2011 41st IEEE International Symposium on Multiple-Valued Logic, pp. 62-68